POLITECNICO DI TORINO
Repository ISTITUZIONALE

OLT(RE)2: an On-Line on-demand Testing approach for permanent Radiation Effects in REconfigurable systems

Original
OLT(RE)2: an On-Line on-demand Testing approach for permanent Radiation Effects in REconfigurable systems / Dario, Cozzi; Sebastian, Korf; Luca, Cassano; Jens, Hagemeyer; Andrea, Domenici; Cinzia, Bernardeschi; Mario, Porrmann; Sterpone, Luca. - In: IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING. - ISSN 2168-6750. - ELETTRONICO. - 6:4(2018), pp. 511-523. [10.1109/TETC.2016.2586195]

Availability:
This version is available at: 11583/2658318 since: 2020-02-27T16:52:50Z

Publisher:
IEEE

Published
DOI:10.1109/TETC.2016.2586195

Terms of use:
openAccess

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

04 May 2023

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

OLT(RE)^2: an On-Line on-demand Testing approach for permanent Radiation Effects in REconfigurable systems

Dario Cozzi, Sebastian Korf, Luca Cassano, Jens Hagemeyer, Andrea Domenici, Cinzia Bernardeschi, Luca Sterpone *Member, IEEE,* and Mario Porrmann, *Member, IEEE*

Abstract—Reconfigurable systems gained great interest in a wide range of application fields, including aerospace, where electronic devices are exposed to a very harsh working environment. Commercial SRAM-based FPGA devices represent an extremely interesting hardware platform for this kind of systems since they combine low cost with the possibility to utilize state-of-the-art processing power as well as the flexibility of reconfigurable hardware. In this paper we present OLT(RE)^2: an on-line on-demand approach to test permanent faults induced by radiation in reconfigurable systems used in space missions. The proposed approach relies on a test circuit and on custom place-and-route algorithms. OLT(RE)^2 exploits partial dynamic reconfigurability offered by today's SRAM-based FPGAs to place the test circuits at run-time. The goal of OLT(RE)^2 is to test unprogrammed areas of the FPGA before using them, thus preventing functional modules of the reconfigurable system to be placed on areas with faulty resources. Experimental results have shown that (i) it is possible to generate, place and route the test circuits needed to detect on average more than 99 % of the physical wires and on average about 97 % of the programmable interconnection points of an arbitrary large region of the FPGA in a reasonable time and that (ii) it is possible to download and run the whole test suite on the target device without interfering with the normal functioning of the system.

Index Terms—Aerospace; On-Demand Test; On-Line Test; Permanent Faults; Radiation Effects; Reconfigurable Systems; SRAM-based FPGAs.

1 Introduction

The interest in employing FPGA-based reconfigurable systems in space applications is growing more and more [1], [2], [3]. Several reconfigurable systems are going to be launched in space in the next few years, e.g., the Polarimetric and Helioseismic Imager Data Processing Unit that will be hosted on the ESA Solar Orbiter [4] (launch planned in 2018) and the Fraunhofer On-Board Processor (FOBP) [5] (launch planned in 2020). These very expensive projects rely on high-end radiation-hardened mature FPGA devices with very high costs and lower performance compared with the latest commercial FPGAs. On the other hand, today's commercial SRAM-based FPGA devices represent the most interesting technological platform for research projects that need high performance and low cost at the same time. Such devices provide a significantly larger amount of resources and they can be dynamically and partially reconfigured, thus allowing performance increase, energy efficiency improvement, and fault tolerance enhancement [6]. Moreover, such devices are significantly cheaper than radiation hardened ones.

Radiations in the atmosphere may damage electronic devices employed in space systems [7], [8]. In particular, radiations may induce both instantaneous and long-term damages. Instantaneous damages are *Single Event Upsets (SEUs)*, i.e., modifications of the content of memory elements in the device, and *Single Event Transients (SETs)*, i.e., transient undesired electrical impulses [7]. Looking at the SRAM-based FPGA technology, SEUs have particularly adverse effects on the configuration memory of such devices, since, by modifying the content of the configuration bits, they may permanently corrupt the structure of the system implemented in the FPGA itself (until a reconfiguration of the device is performed) [9]. The long-term damages induced by radiations on electronic devices are caused by the *Total Ionizing Dose (TID)*, i.e., the accumulation of charge trapped in the oxide layer of transistors in CMOS circuits [8]. TID first causes a degradation of the performance of the system, in terms of timing and power consumption, because of threshold voltage shift and leakage current increase, and ultimately it may also cause the complete destruction of

This work has been partially supported by the ESA (European Space Agency) Innovation Triangle Initiative (ITI) – Project A00016022 "On-Line Testing and Healing Permanent Radiation Effects in Reconfigurable Systems" and by the Cluster of Excellence Cognitive Interaction Technology 'CITEC' (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG). The authors also personally thank David Merodio Codinachs and Jørgen Ilstad from ESA (European Space Agency).

Dario Cozzi, Sebastian Korf, Jens Hagemeyer and Mario Porrmann are with the Center of Excellence Cognitive Interaction Technology, Bielefeld University, Germany. e-mail: {dcozzi, skorf, jhagemey, mporrmann}@cit-ec.uni-bielefeld.de

Luca Cassano is with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italy. e-mail: luca.cassano@polimi.it

Cinzia Bernardeschi and Andrea Domenici are with the Dipartimento di Ingegneria dell'Informazione, Universitá di Pisa, Italy. e-mail: cinzia.bernardeschi@ing.unipi.it

Luca Sterpone is with the Dipartimento di Automatica ed Informatica, Politecnico di Torino, Italy. e-mail: luca.sterpone@polito.it

Manuscript received MMMM DD, YYYY; revised MMMM DD, YYYY.

Copyright (c) 2015 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending an email to pubs-permissions@ieee.org.

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

parts of the system, and thus functional failures.

In the area of fault detection in FPGA devices, a lot of work has been done in the last two decades, addressing hardware structural defects due to the manufacturing process [10], [11], [12], [13]. More recently, some work about detecting SEUs in the configuration memory has been published [14], [15], [16]. On the other hand, the problem of detecting faults induced in FPGA devices by the long-term exposure to radiation has not yet exhaustively been explored. In the related work section the advantages and drawbacks of several approaches related to fault detection in FPGA devices will be discussed in details.

In this paper we present OLT(RE)^2: an on-line on-demand approach to test permanent faults induced by radiations in the routing resources of SRAM-based FPGAs that are generally employed in reconfigurable systems used in space missions. The proposed approach relies on a test circuit, composed of a *Test Pattern Generator (TPG)* and an *Output Response Analyzer (ORA)*, to test the physical wires and the Programmable Interconnect Points (PIPs) located between the TPG and the ORA, i.e., the *Nets Under Test (N-UTs)*. Moreover, the approach exploits a new place-and-route algorithm, named *U-TURN*, specifically designed to maximize the amount of routing resources tested by each test circuit. Once the test circuits have been mapped, placed-and-routed, they are stored and they can then be used for the on-line on-demand testing activity. In particular, the testing procedure is meant to be launched before a functional module is placed on the FPGA, i.e., it is not intended for testing already occupied reconfigurable areas. More in details, the OLT(RE)^2 approach exploits partial dynamic reconfiguration to place the test circuits at run-time on the free areas of the FPGA to test them before using these areas to place the functional modules when required. In this way it is possible to prevent the functional modules of the reconfigurable system to be placed on faulty areas of the FPGA at run-time. OLT(RE)^2 aims at helping designers in making the use of high-performance state-of-the-art commercial SRAM-based FPGAs viable for space applications. Moreover, OLT(RE)^2 could help in those scenarios, such as low-cost academic micro- and nano-satellites, where the high cost of radiation hardened FPGAs is unaffordable.

The basic idea underlying OLT(RE)^2 has been proposed in a preliminary version (before any implementation was performed) in [17]. The structure of the test circuits and a preliminary experimental evaluation has been presented in [18]. W.r.t. the previously published work, the current paper presents the completely new place-and-route algorithm used to place and route the test circuits in the design phase; moreover, the complete OLT(RE)^2 CAD flow is presented; finally, an extensive experimental evaluation of the proposed approach on a wide range of FPGA families (Spartan-6 and Virtex-4, -5 and -6) is discussed. In this paper, we focus on the detection of faults before a functional module is placed on a reconfigurable region utilizing partial dynamic reconfiguration. Methods for the additionally required fault mitigation of permanent faults have been proposed in [17] utilizing patching macros.

The remainder of this paper is organized as follows: Section 2 reviews the published research work related to testing of FPGAs and reconfigurable systems; Section 3 presents the main terms used in the paper; Section 4 discusses the effects of radiations on FPGAs; Section 5 presents the OLT(RE)^2 approach, the test circuits, the place-and-route algorithm and the OLT(RE)^2 CAD flow; Section 6 present the experimental results achieved by the proposed testing approach and, finally, Section 7 concludes the paper.

2 Related Work

The approaches to FPGA testing can be classified in two families: *application-independent* and *application-dependent* ones.

Application-independent methods, such as [10], [11], [12], [13], are meant to detect structural defects due to the manufacturing process of the chip. These approaches are intended to be performed off-line, by the FPGA manufacturer, and they target every possible fault in the architecture without any consideration of which parts of it are actually used by a given design. These techniques generally employ multiple test configurations and the associated ad-hoc generated test patterns. Each test configuration is intended to test a subset of the possible faults of the chip.

On the other hand, application-dependent methods, such as [14], [15], [16], [19], focus only on those resources of the FPGA actually used by a given system. For this reason, these techniques are meant to be applied by the end-user of the FPGA device, either off-line or on-line, after the system design has been defined. The rationale behind these techniques is that an FPGA-based system occupies only a subset of the resources provided by the FPGA device. Therefore, it is sufficient to demonstrate that the resources used by the implemented system are fault-free to guarantee the correct operation of the system itself. Application-dependent methods have been proposed for in-service testing of both structural defects [14], [15], [19] and SEUs [16].

Permanent faults caused by the TID have not yet been extensively addressed by testing techniques. In the last years, the ongoing shrinking of the feature size of the CMOS technology made SEUs the predominant radiation effect in electronic devices. Thus, researchers focused much more on the detection of SEU effects than on TID. Nevertheless, the significant voltage scaling and the dramatic increase of the number of transistors make TID again significant in modern space electronics [20]. To the best of our knowledge, the only work addressing the problem of detecting faults induced in FPGA devices by the long-term exposure to radiation is the one reported in [21]. This work focuses on the architecture used for testing reconfigurable areas inside an FPGA device, i.e., the Configurable Logic Blocks (CLBs) and on the scheduling of the test activities. On the other hand, little information about how the test circuits and associated test patterns are generated is given. Moreover, the presented test architecture addresses any possible faults in those resources of the FPGA device actually used by the implemented design, leaving out the remaining resources.

It is worth noting that manufacturing defects and radiation-induced permanent faults have very similar effects under a functional point of view. Indeed, in both cases open and short faults may be observed during the functioning of the system. Therefore, many ideas belonging

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

to application-independent testing may be borrowed and reused by approaches aiming at detecting permanent effects of radiations. As it has been discussed above, application-independent techniques exploit multiple configurations of the FPGA, each one intended to test a subset of the resources available in the device. These test configurations are generally composed of a *Test Pattern Generator (TPG)* that provides input stimuli to the set of resources under test and of an *Output Response Analyzer (ORA)* that observes the output of the resources under test and determines whether they are faulty or not. These techniques may additionally be divided into two sub-categories: *Comparison-based* and *Parity-based* [22].

In the Comparison-based techniques, such as the ones presented in [13], [23], the ORA knows both the expected output (associated with the input stimuli generated by the TPG) and the actual output produced by the resources under test and by comparing them it is able to determine whether a fault occurred in the system. The main drawback of these techniques is that they are not able to detect faults in the TPG and those faults that do not interfere with the actual output of the resources under test, i.e., *equivalent faults*. Parity-based testing techniques, such as [24], [25], have been introduced to overcome these limitations. Here, the TPG additionally calculates a parity bit on its outputs; in turn, the ORA calculates the parity bit on the received signals. By comparing these two parity bits the ORA is able to detect the occurrence of a fault in the resources under test. In this way the ORA does not need to know the expected output but it only relies on the parity bit produced by the TPG, therefore, also faults affecting the TPG and the equivalent faults may be detected.

Parity-based testing approaches may be additionally classified in two families: *single parity* [24] and *cross-coupled parity* [25]. In the single parity-based techniques, the TPG is an n-bit counter, producing n+1 output bits (the last bit is the parity bit calculated on the previous n bits). The ORA receives n+1 input bits, calculates the parity on the first n bits and compares it with the received parity bit. The drawback of single parity-based techniques is that some faults in the TPG and some equivalent faults may still not be detected by the ORA. Moreover, it is vital that the parity bit is sent on a fault-free wire. In cross-coupled parity-based techniques the TPG is composed of two independent n-bit counters, let us call them `TPG_a` and `TPG_b`; each TPG produces n output bits plus one parity bit. Similarly, the ORA is duplicated: `ORA_a` receives the n input bits from `TPG_a` and the parity bit from `TPG_b`; conversely, `ORA_b` receives the n input bits from `TPG_b` and the parity from `TPG_a`. In this way all the faults occurring in the TPGs and all the equivalent faults may also be detected.

OLT(RE)^2 falls in the category of on-line application-independent cross-coupled parity-based testing techniques. OLT(RE)^2 provides the basic building blocks for an on-line on-demand testing procedure for reconfigurable systems: the test circuit and the place-and-route algorithm needed to exploit them.

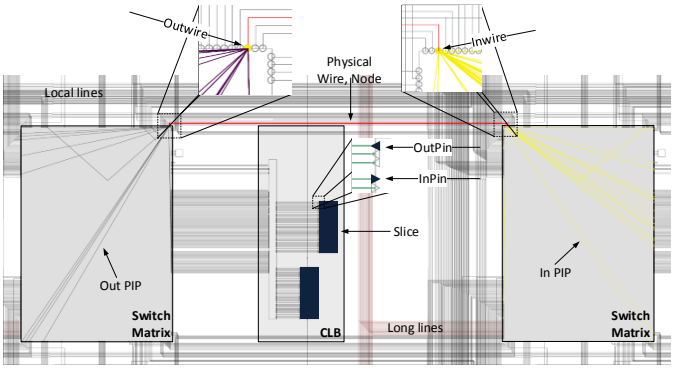

Fig. 1. A summary of the FPGA resources terminology.

3 Terminology

In this section we provide some basic terminology that will be extensively used in the rest of the paper. We refer to Figure 1 (a screenshot of a Xilinx FPGA, taken from the Xilinx FPGA Editor tool) that shows a portion of an FPGA, highlighting the main components considered in the OLT(RE)^2 flow, that are:

- *Slice*: the basic logic building block of an FPGA. A slice includes the configurable resources for the implementation of Boolean functions, as well as flip-flops and carry-propagation logic.
- *Configurable Logic Block (CLB)*: a group of several slices and interconnection resources.
- *Switch Matrix (SM)*: the basic configurable interconnection component in the FPGA architecture. A SM allows the designer to define the routing of the signals within the FPGA.
- *Physical Wire (PW)*: a hardwired non-configurable interconnection between either two switch matrices or a switch matrix and a slice.
- *Pin*: a connection point between a slice and a PW. In case the direction of the signal goes into the slice, the Pin is called *InPin*; *OutPin* otherwise.
- *Wire*: a connection point between a PW and a SM. In case the direction of the signal goes into the SM, the Wire is called *InWire*; *OutWire* otherwise.
- *Programmable Interconnection Point (PIP)*: a configurable connection between two wires belonging to the same SM. It is worth noting that multiple PIPs are connected to the same InWire as well as multiple PIPs are connected to the same OutWire.

4 Effects of Radiation on SRAM-based FPGA devices

As discussed in the introduction, radiations in space may cause both instantaneous (SEUs and SETs) and long-term (TID) damages in electronic devices. Since the OLT(RE)^2 approach is specifically intended to detect permanent faults, in the following we leave out the discussion about SEUs and SETs, referring the reader to more specific papers [7].

The Total Ionizing Dose (TID) is the effect of the accumulation of the charge injected by radiation in the oxide layer of transistors in CMOS circuits. The amount of accumulated

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

charge obviously depends on the exposure time, the flux of the particles and their Linear Energy Transfer (LET), i.e., the amount of energy that ionizing particles transfer to the material traversed per unit area. TID first causes a global decrease of the device performance: transistors slow down (due to the voltage threshold shift) and the power consumption increases (due to increased leakage current). A secondary effect of TID is the increase of the SEU and SET sensitivity of the circuit. Finally, TID may make regions of the device totally unusable, thus causing permanent faults. When looking at SRAM-based FPGAs, an additional effect of the transistor's threshold voltage shift caused by TID is the loss of reconfigurability of the device [8].

In modern FPGAs, the routing resources represent up to 90 % of the whole chip area [26]. Therefore, the OLT(RE)^2 approach focuses on the detection of faults occurring in the routing architecture of FPGA devices. When a permanent fault occurs in an FPGA device, the fault may affect either the routing resource itself, i.e., the fault directly damages physical wires, or the configuration resources associated with routing resources, i.e., the fault affects PIPs. In the former case the functional effect of the fault is a stuck-at 0/1, i.e., the signal on the wire is stuck at the logic value 0 or 1. In the latter case the functional effect of the fault is a stuck-off/on, i.e., the PIP remains unprogrammed/programmed [22].

Other faults may affect the routing structure of an FPGA device, i.e., bridges, breaks on wires and permanent faults in pass transistors. Nevertheless, we argue that these faults typically appear directly after the manufacturing of the device and thus, they may be detected through post-manufacturing tests executed by the chip manufacturer. Since the proposed approach is meant to be applied at runtime, we do not consider these faults.

5 The OLT(RE)^2 Test Approach

5.1 Overview of the Proposed Approach

The overall goal of OLT(RE)^2 is supporting on-line on-demand testing of dynamically reconfigurable systems based on SRAM-based FPGAs. More in details, OLT(RE)^2 exploits dynamic partial reconfiguration capabilities offered by modern SRAM-based FPGA devices to place previously ad-hoc designed, placed-and-routed test circuits on the reconfigurable areas of the system. These test circuits are meant to be placed and run before a functional module of the reconfigurable system has to be placed, in order to verify whether the area where the functional module will be placed is faulty or fault-free, thus avoiding to place the functional modules on faulty resources of the device and globally increasing the reliability of the reconfigurable system.

From a very high-level point of view, the OLT(RE)^2 approach can be summarized in the following steps:

- a test circuit is designed, composed of a test pattern generator, an output response analyzer and several nets under test;
- the test circuit is placed-and-routed multiple times (thus each placed-and-routed test circuit will test a subset of the routing resources of the Area Under Test) in such a way as to maximize the amount of resources under test while minimizing the number of test circuits (and thus the number of reconfigurations required at run-time);
- place a test circuit at a time on the Area Under Test. Once the test circuit is placed, there is no interaction between it and the IOBs. Indeed, the test circuits internally generate test stimuli that are propagated to the resources under test and then received and analysed by an output response analyzer that determines whether the resources under test are faulty or not;
- Finally, the result of the test is stored in (distributed) memory elements that are accessed through memory readback. By knowing the resources occupied by the failed test circuit it is also possible to identify a subset of possible faulty resources, thus performing a coarse-grained fault diagnosis.

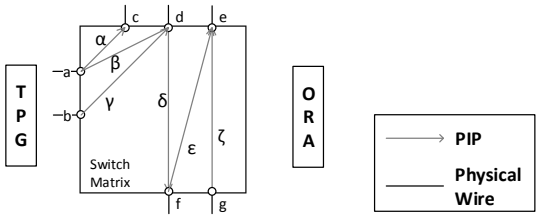

Fig. 2. An example switch matrix.

It is worth noting that the proposed testing approach focuses on the majority of the routing resources available in an FPGA device. The only routing resource that are currently not supported by OLT(RE)^2 are the routing resources connected to DSPs, the block RAMs, the carry chains and the clock distribution resources.

Finally, it has to be mentioned that OLT(RE)^2 is suitable for a wide range of Xilinx FPGA families, including Spartan-6, Virtex-4, -5 and -6. Due to its modular structure, OLT(RE)^2 can be extended to work with other Xilinx FPGA families. Additionally, the methodology can be utilized for FPGAs from other vendors.

5.2 Routing Faults Test Principles

In order to better understand the proposed approach, we will first recall concepts related to fault detection in the routing resources of an FPGA that can be found in the literature [22]. Let us consider as an example the switch matrix shown in Figure 2. The switch matrix has 6 PIPs (namely α, β, γ, δ, ϵ and ζ) and 7 PW (namely *a, b, c, d, e, f* and *g*). Let us call *PIP-UT* the PIP under test and *PW-UT* the physical wire under test.

When the target is on stuck-at faults on physical wires or stuck-off faults on PIPs, the PW-UT and the PIP-UT are the first PW and the first PIP to which the output of the TPG will be connected. The rationale behind this is that it is enough to use a routing resource and to write on it both a logic 1 and a logic 0 to be able to detect stuck-at 0/1 and stuck-off faults. Therefore, looking at Figure 3, we see that the N-UT crosses PW *a* and PW *d* and PIP β. Therefore, the N-UT will detect stuck-at 0/1 faults on *a* and *d* and stuck-off faults on β.

On the other hand, when the target is on stuck-on faults on PIPs, two nets under tests are needed, each connected to

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

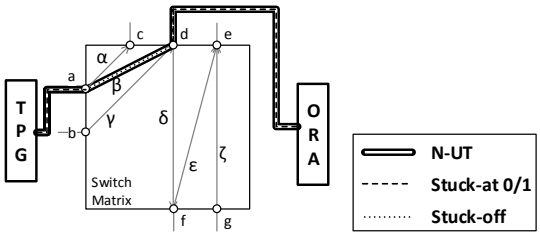

Fig. 3. An example routing between TPG and ORA for stuck-at/stuck-off testing.

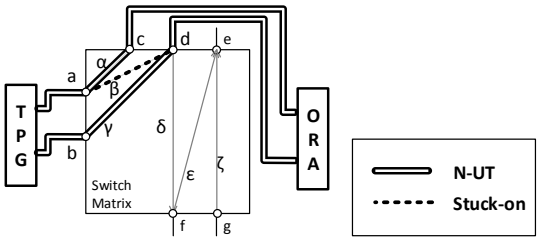

Fig. 4. An example routing between TPG and ORA for stuck-on testing.

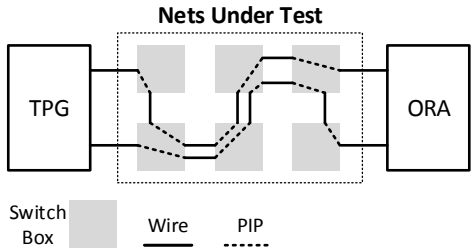

Fig. 5. The high-level representation of a test circuit.

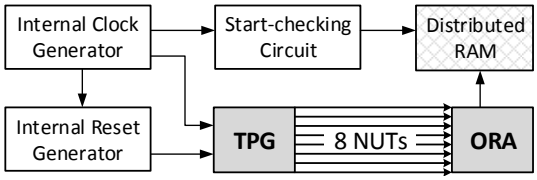

Fig. 6. The structure of a test circuit.

one of the two ends of the PIP-UT (as shown in Figure 4, where the target is a stuck-on on β). It is worth noting that in this case none of the two N-UTs actually uses the PIP-UT. The idea behind this is that a stuck-on fault on a PIP may cause a short between two nets. Thus, by routing two N-UTs through the two ends of the PIP-UT we actually check whether the two nets are shorted (and thus the PIP-UT is stuck-on affected) or not. Note that, apart from the stuck-on fault on β, the two N-UTs also test stuck-at 0/1 faults on *a*, *b*, *c* and *d* and stuck-off faults on α and γ.

The principles discussed above have been integrated in U-TURN, used to place-and-route the test circuit.

5.3 The Test Circuit

The test circuit on which OLT(RE)^2 relies (see Figure 5 for a very high-level representation) is composed of a TPG and an ORA. All the connections between the outputs of the TPG and the inputs of the ORA represent the resources under test. More specifically, all the physical wires and all the PIPs connecting TPG and ORA represent the nets under test (N-UTs). In the current version of the test circuit (originally designed for the Xilinx Virtex-4 family of devices, that provides 4-input look-up tables) the test circuit has 8 N-UTs. When designing the test circuit we defined the following goals:

- Detect 100 % of the faults in the routing resources of the Area Under Test;
- Detect the largest possible amount of faults in the resources occupied by the test circuit itself;
- Occupy the smallest possible amount of resources of the FPGA, in order to be applicable also when a large part of the resources of the FPGA is already occupied.

The functional block-level structure of the designed test circuit is depicted in Figure 6: the clock and reset signals that are fed to the test circuit are generated by dedicated modules, in order to make the testing structure entirely independent of the region of the FPGA on which it is placed. No external clock and reset signals are used and no input/output buffers are employed. Therefore, it is possible to change the Area Under Test by only re-placing the test circuit, without any change in the logic. The result of the test is stored in dedicated LUTs configured as distributed RAM, whose content can be read-back at the end of the testing activity. A ring oscillator is used to generate the internal clock signal. A parametric n-bit shift register pre-loaded with n 1s is used to generate the reset signal. The start-checking circuit has been added to the test circuit to verify whether the test circuit has been correctly configured and the test correctly started. Some faults may prevent the test to start at all: in such a case, although a fault actually occurred, the ORA would not be able to detect any misbehaviour.

In more detail, the TPG is composed of two 2-bit counters, each placed into two 4-input LUTs and using two flip-flops (thus, each counter occupies only one slice). One of the two counters is an *up counter*, i.e., counts from 0 up to 3, and produces an even parity bit; the other counter is a *down counter*, i.e., counts from 3 down to 0, and produces an odd parity bit. The ORA is composed of two analyzers, one for each counter, each occupying one 4-input LUT (thus, the whole ORA occupies only one slice). By comparing the received 2-bit state with the received parity bit each analyzer is able to determine whether a fault occurred in its input wires or in the associated counter. We point out that we verify the correctness of the used DRAM before placing the testing circuit by writing both 0/1 and reading back the values. The detailed structure of the testing circuit is depicted in Figure 7.

In the remainder of the paper, for the sake of simplicity, we refer to placement-and-routing of TPG, ORA and N-UTs, when referring to the placement-and-routing of the whole test circuit.

5.4 The U-TURN Place-and-Route Algorithm

The U-TURN algorithm represents the core of the OLT(RE)^2 testing approach. The basic idea is that the previously

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

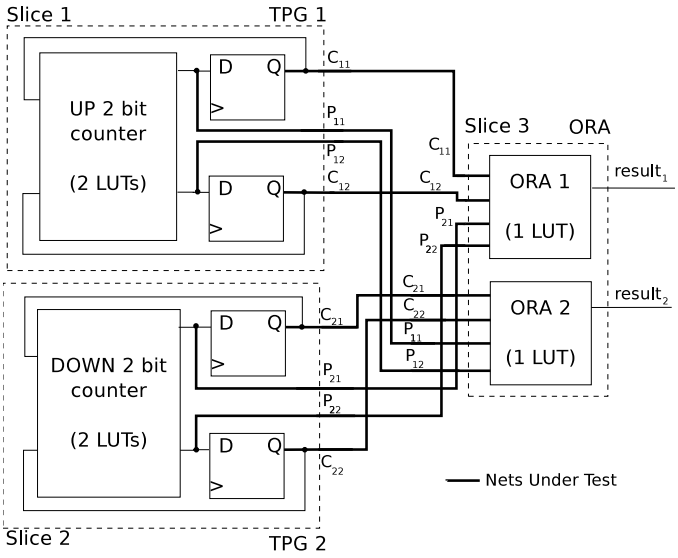

Fig. 7. The detailed structure of the testing circuit.

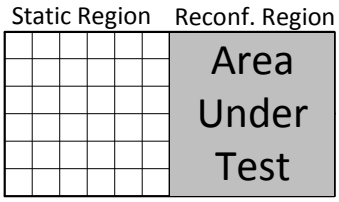

Fig. 8. The Area Under Test.

```
1: Function U-TURN()
2: testedResources ← ∅
3: divide the Area Under Test into n sub-areas under test
4: for each Sub-Area Under Test a_i do
5:    Place TPG and ORA
6:    for each PIP pip_ut in a_i do
7:       testingCircuit ← ∅
8:       for each output tpg_j of the TPG do
9:          sm_ut ← getSwitchMatrix(pip_ut)
10:         route tpg_j to pip_UT according to the targeted fault
11:         nut ← N-UT_Router(pip_ut, 0)
12:         testedResources.add(getAllUsedResources(nut))
13:         route nut to input j of the ORA
14:         testingCircuit.add(nut)
15:      end for
16:      save testingCircuit
17:   end for
18: end for
19: EndFunction
```

Algorithm 1: The overall placement and routing algorithm.

presented test circuit, composed of a TPG, an ORA and 8 nets under test is placed once and then the N-UTs are routed multiple times (through multiple iterations of the U-TURN algorithm). By changing the routing of the N-UTs (and leaving unaltered the placement of TPG and ORA) it is possible to test different resources in the Area Under Test. Each run of U-TURN will thus generate a placed-and-routed test circuit that will cover a given subset of the routing resources available in the Area Under Test. In particular, after a set of dedicated experiments, we discovered a technological constraint on the Xilinx architectures: each single net (and thus also each network under test in our approach) can occupy (and test) no more than 100 PIPs, and thus no more than 101 physical wires. The constraint of 100 PIPs was chosen as a result of experiments concluded on different Xilinx devices and families. When utilizing too many PIPs in one net, the signal on the particular net shows accumulated jitter and slew rate degradation, eventually resulting in failing circuit functionality. The effect is depending on device family, temperature, supply voltage, and process variations. Choosing a limit of 100 PIPs guarantees a stable test circuit operation across all device families and environment conditions. In the following, we will introduce the TPG and ORA placement algorithm and the N-UTs routing algorithm.

5.4.1 The TPG & ORA Placer

In a typical reconfigurable system, an FPGA is partitioned in a static region and a reconfigurable region. The reconfigurable region is the area where functional modules can be dynamically placed; therefore the content of the reconfigurable region may change at run-time. The static region is the area of the FPGA where all the structures required to support the dynamic reconfiguration of the reconfigurable region are placed; thus, the content of the static region is fixed. OLT(RE)^2 focuses on faults in the routing resources of the reconfigurable region (since it relies on partial dynamic reconfiguration), while the static region has to be tested with dedicated approaches.

Let us assume the FPGA in Figure 8 as an example where the right-bottom dark-grey box represents the Area Under Test. Let us also refer to Algorithm 1 that presents the pseudo-code of the placement algorithm. The first step that the algorithm performs is the division of the Area Under Test into N non-overlapping sub-areas under test (line 3 of Algorithm 1). The testing procedure will focus on one of the obtained sub-areas at a time. When testing a given sub-area, TPG and ORA are placed in one of the remaining sub-areas under test (line 5 of Algorithm 1); in this way TPG and ORA do not occupy resources belonging to the sub-Area Under Test, that can therefore be entirely tested. Moreover, it is worth noting that TPG and ORA are always placed in one of the sub-areas under test (and thus always into the Area Under Test itself): in this way we ensure that the testing procedure does not interfere with the functioning of the modules placed outside the Area Under Test (see Figure 9 for an example of partitioning of the Area Under Test and placements of TPG and ORA).

After placing TPG and ORA (which will remain in the same position for the test of an entire sub-area), the U-TURN algorithm has to be iterated multiple times in order to generate all the test circuits needed to entirely cover the Sub-Area Under Test. For each test circuit, each output of the TPG has to be routed through the Area Under Test (line 10 Algorithm 1) and then to an input of the ORA (line 13 Algorithm 1). It is worth noting that, although each test

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

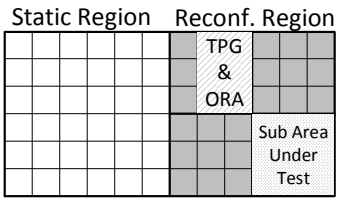

Fig. 9. An example partitioning of the Area Under Test and placement of TPG and ORA.

circuit covers multiple resources, the generation process of a test circuit focuses on one PIP, the PIP under test (PIP-UT, pip_{ut} at line 6 of Algorithm 1). The connection between the output of the TPG and the starting point of the N-UT depends on which fault the N-UT is meant for (as discussed in the previous section).

5.4.2 The N-UTs Router

The N-UT router is one of the key component of the OLT(RE)^2 approach. Once the output of the TPG has been routed to the PIP-UT (line 10 of Algorithm 1), the recursive `N-UT_Router` algorithm is run to build the N-UT (line 11 of Algorithm 1). The pseudo-code of `N-UT_Router` is shown in Algorithm 2. The N-UT will exit from the switch matrix sm_{ut} through PIP pip_{ut}, and it will reach a SM sm_k through PW pw_j and it will finally occupy one of the PIPs reachable from pw_j (let us call this PIP pip_x). All the resources that the N-UT occupies are stored in a temporary solution as well as marked as already visited (line 11 of Algorithm 2). Each time the N-UT comes back to the switch matrix sm_{ut} the *check-Solutions()* procedure is invoked (line 14 of Algorithm 2). This procedure checks whether the temporary solution occupies more yet untested resources than the current best solution. If yes, the temporary solution becomes the best solutions, while no exchange is performed otherwise. Given the technological constraint discussed above, the procedure is repeated until the N-UT occupies 100 PIPs. Finally, when the N-UT construction process is completed, the N-UT is routed from the PIP-UT to one of the inputs of the ORA (line 13 of Algorithm 1). Note that, as we previously said, even if each N-UT targets one PIP, after being routed, the N-UT covers multiple physical wires and PIPs, thus the test circuit generation procedure needs always to keep track of the already covered PIPs (line 12 of Algorithm 1) in order to avoid considering these as PIP-UT of the following N-UTs, and thus saving computational time.

It is fundamental to point out that U-TURN relies on the database of architectural resources described in [27]. This database contains a compressed (some MBs) but complete image of the FPGA device architecture extracted from the Xilinx XDL representation of the entire device (several GBs). The information stored in the database is used to actually drive the U-TURN algorithm.

5.5 The OLT(RE)^2 CAD Flow

We implemented a set of C++ tools and we integrated them into the standard Xilinx CAD flow in order to automate all the activities that need to be carried out to implement the proposed test strategy. The OLT(RE)^2 CAD flow (see

```
1: Function N-UT_Router(pip_ut, nUsedPIPs)
2: if nUsedPIPs > 100 then
3:    return
4: end if
5: get the physical wire pw_j reachable from pip_ut
6: for each PIP pip_x not yet visited and reachable from pw_j
   do
7:    get the physical wire pw_y reachable from pip_x
8:    if pw_y has not yet been visited then
9:       add pip_x and pw_y to the temporary solution
10:      set pip_x and pw_y as visited
11:      if pip_x belongs to sm_ut then
12:         checkSolutions()
13:      end if
14:      N-UT_Router(pip_x, nUsedPIPs+1)
15:      remove pip_x and pw_y from the temporary solution
16:   end if
17: end for
18: return
19: EndFunction
```

Algorithm 2: The N-UT creation algorithm.

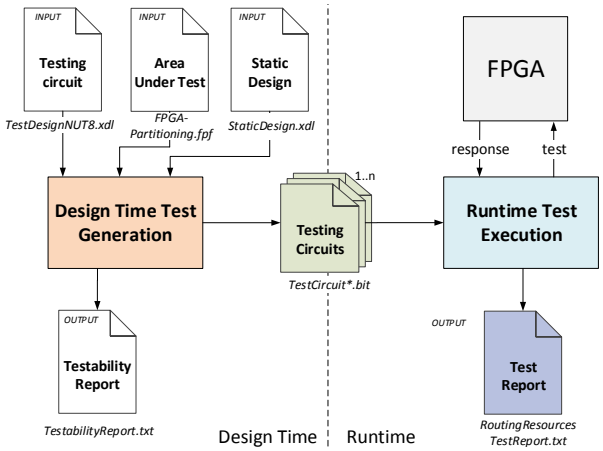

Fig. 10. The overall OLT(RE)^2 CAD flow.

Figure 10) can be divided in two parts: the *design-time test generation* and the *run-time test execution* sub-flows. The design-time test generation is meant to be performed at design-time on a ground machine to generate all the test circuits. The run-time test execution, on the other hand, is meant to be executed at run-time, on the reconfigurable system itself, before each reconfiguration.

The flow utilizes the XDL intermediate language. Moreover, it utilizes the typical PR flow of Xilinx; it takes a testing circuit design (mapped with the Xilinx tool), places and routes it, and generates a certain number of testing circuits (in XDL format). Finally, the testing circuits are converted in the Xilinx NCD format and can be integrated in the normal Xilinx flow.

5.5.1 The design-time test generation sub-flow

Figure 11 depicts the design-time test generation sub-flow. The input files consist of a specification of the partitioning of the system (a .fpf file) that specifies the region(s) under test (RUTs), the design of the static region and the test circuit

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

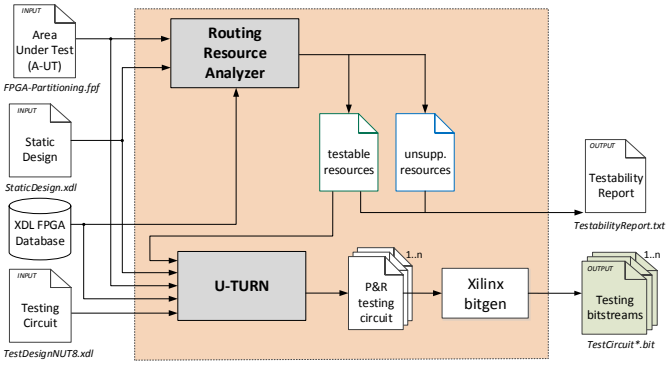

Fig. 11. The design-time test generation sub-flow.

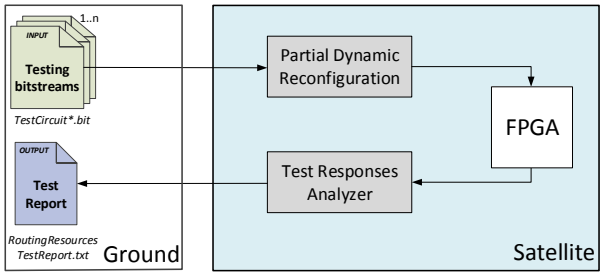

Fig. 12. The run-time test execution sub-flow.

design (a .xdl file containing the specification of the test circuit described in Section 5.3).

The first tool, dubbed *Routing Resources Analyzer (RRA)*, divides the routing resources (physical wires and PIPs), for each given RUT, in *unsupported*, i.e., the routing resources related to BRAMs, DSPs, carry chains and clock distribution, and *testable*, i.e., all the remaining routing resources. Moreover, by reading the design of the static region the RRA detects and excludes all the routing resources in the reconfigurable region that belong to the static region. In this way, the U-TURN algorithm can focus on the sole testable resources, thus saving time avoiding testing unsupported ones. At the end of this process, a detailed report of the testability of the design is created, which lists all routing resources with their assigned categorization.

The output of RRA contains crucial information for U-TURN. The execution of U-TURN produces multiple placed-and-routed test circuits, which cover all testable routing resources for the RUTs. To carry out this task, three input files are needed: the list of testable resources, the partitioning file and the test circuit specification. The resulting placed-and-routed test circuits are specified in XDL and subsequently translated into a Xilinx design file (NCD) or optionally into a Xilinx hard macro file (NMC) and then into a (partial) bitstream with the standard Xilinx tools (xdl and bitgen).

5.5.2 The run-time test execution sub-flow

Figure 12 depicts the run-time test execution sub-flow. This sub-flow is executed as soon as a reconfiguration of the system is needed. The test circuits (either stored in a dedicated fault tolerant persistent memory or received "just in time" from the ground station) are exhaustively placed on the area under test one at a time and the test is executed. After each test circuit has been properly placed and run, the ORA stores the result of the test execution in a dedicated distributed memory. These results are read back by the *Test Analyzer* that cumulatively creates the overall test report.

6 Experimental Results

We performed three sets of experiments in order to evaluate the correctness, effectiveness, and efficiency of the proposed testing approach. First of all, we were interested in verifying that the designed basic test circuit (without considering the U-TURN place-and-route algorithm) was actually able to detect all the possible faults that may occur in the N-UTs. Then, we wanted to assess the actual fault coverage that OLT(RE)^2 allowed achieving with the generated placed-and-routed test circuits as well as its efficiency in terms of time required to generate the test circuits and of size of the test circuits. Finally, we were also interested in analyzing the run-time efficiency of the approach in terms of time needed to transfer the test circuits to the device under test and then run the entire test. In the following of the section we report about these points.

6.1 Test Circuit Validation

As a first experiment we have validated the implemented basic test circuit. More in details, we were interested in assessing whether the circuit was actually able to detect all the faults that may occur in the nets connecting the TPG and the ORA.

In order to do so, we have emulated the occurrence of faults in the NUTs. In more detail, we have manipulated the XDL file of the testing circuit to reproduce the effect of faults. We injected stuck-on/off faults by activating/removing PIPs in the XDL file; similarly, we injected stuck-at 0/1 by forcing the output of specific LUTs in the TPG to 0/1. Finally, the XDL file is translated into a "faulty" bitstream and downloaded in the FPGA device.

The fundamental result of this preliminary validation campaign is that the designed test circuit is able to detect 100 % of the faults occurring in the N-UTs. Additionally, 100 % of the faults occurring in the routing resources occupied by the Internal Reset Generator and by the Distributed RAM are detected. The great majority (about 97 %) of the faults occurring in the routing resources occupied by the Internal Clock Generator component are also detected. The only critical sub-component of the test circuit is the Start-checking Circuit: When faults occur in the routing resources occupied by this sub-component, the test fails (because the Start-checking Circuit does not recognize the start of the test) even if the N-UTs are not affected by faults.

In Table 1 the amount of resources (number of LUTs used for logic, distributed RAM and shift-registers and number of used flip-flops as well as the total number of occupied slices) occupied by the test circuit (i.e., TPG and ORA) for the four currently supported FPGA families (Spartan-6 and Virtex-4, -5 and -6) are summarized. The area occupation in terms of percentage of used resources actually depends on the specific device: for example, the area occupation on

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

TABLE 1
Area occupation of the designed test circuit for several FPGA families.

Family	#LUTs			#FFs	#slices
	logic	memory	shift reg.		
Virtex-4	35	3	1	11	24
Virtex-5	25	3	1	11	18
Virtex-6	25	3	1	10	20
Spartan-6	33	3	1	13	23

the smallest Virtex-4 device, the FX12 (counting 5472 slices) is only 0.43 %; of course, the percentage of area occupied in larger devices will be even smaller. Thus, we argue that, as expected, the proposed testing technique could be applied also in case the majority of the FPGA area is already occupied.

6.2 Design-time Performance Analysis

The second analysis aimed at assessing the performance of the design-time test circuit generation. We present the routing resource fault coverage, the time needed to place-and-route all the required test circuits for various FPGA families and devices first in order to assess the effectiveness of the approach, and then for various sizes of the area under test on the same device, in order to assess scalability.

6.2.1 Experimental Setup

We executed the Design-time test generation flow on eight devices belonging to four different families (Spartan-6 and Virtex-4, -5 and -6). In this way, we ensure that OLT(RE)^2 can be utilized on a wide range of FPGAs families. In the effectiveness analysis, one clock region was considered as Area Under Test. When focusing on scalability, we employed a Virtex-4 XC4VFX12 device and we considered areas of one clock region up to four clock regions as Area Under Test. The U-TURN algorithm was launched on a PC equipped with an Intel Xeon Processor W3565 with 24 GB of RAM. After the generation of the test circuits by U-TURN, we assessed the achieved fault coverage by measuring the amount of the routing resources of the Area Under Test occupied by the whole set of test circuits. To do so, we interacted with the Xilinx FPGA Editor tool and we exploited its internal representation of the Xilinx devices.

6.2.2 Effectiveness Analysis

We measured the fault coverage achieved by OLT(RE)^2 for several device families and models. Table 2 reports the results of this first experiment.

For each device family (reported in the first column) we analysed two devices (second column): a small- and a large-size one. For each device, columns three, four, and five report the number of possible stuck-at faults on physical wires and the number of possible stuck-off and stuck-on faults on PIPs, respectively; finally, the last three columns report the achieved coverage for the stuck-at, stuck-off and stuck-on faults, respectively.

It can be observed that the proposed testing technique always achieves a high fault coverage (more than 98 % in most cases) both for stuck-at faults on physical wires and for stuck-off and stuck-on faults on PIPs. It is worth noting that these good results are achieved for all the device families, which are different in terms of architecture.

6.2.3 Efficiency and Scalability Analysis

Table 3 reports about the efficiency of the proposed approach. More in details, for each device the table shows the number of test circuits generated to achieve the fault coverage values reported in Table 2 as well as the total size (in MB) of the test bitstream suite and the size of the test bitstream suite after bitstream compression and, finally, the time required at design time to generate them.

A first consideration that can be drawn from the results shown in Table 3 is that the time required at design time to generate the test bitstream suite is quite reasonable, ranging from about 3 hours for the smallest device up to about 4 days for the largest one; the generation of the test bitstreams has to be performed only once at design-time. Moreover, since the test circuit and the overall approach are application-independent, it is worth noting that, after having been generated for a given FPGA device, the test bitstreams can be used for several reconfigurable systems using the same device without any modification.

If we look at the size of the test bitstreams we can see that the "raw" size is very large, ranging from hundreds of MBs up to some GBs. This may represent a limitation of the proposed testing technique, since, as we previously discussed, test bitstreams have to either be stored in an on-board persistent memory, or received "just in time" from the ground station. This problem can largely be alleviated by using a compression algorithm to reduce the size of the bitstream. Even if bitstream compression was not in the scope of the paper, we investigated how much the total size of the bitstream could be reduced. It can be observed that we reduced the overall suite size to about 10 % of the original size with a simple run of the ZIP algorithm.

Of course, more sophisticated, ad-hoc, bitstream compression algorithms may further increase the compression ratio [28]. Another way to optimize the number and the size of the partial bitstreams is *bitstream relocation* that, thanks to the homogeneity of Xilinx FPGAs, allows multiple partial bitstreams to be generated starting from an "original" one and only changing its address in the configuration memory [29], [30]. Finally, test performance could be improved by placing several testing circuits in parallel. On the other hand, this could introduce the problem of fault masking.

As an additional experiment we were interested in assessing the scalability of OLT(RE)^2. To do so, we performed multiple experiments in which we applied the approach to a Virtex-4 device and we increased the size of the Area Under Test from 1 clock region up to 4 clock regions (representing half of the entire device).

Table 4 reports the results of this scalability analysis experiment. More in details, the first four columns report the size of the Area Under Test in terms of number of clock regions, the number of stuck-at faults on physical wires and the number of stuck-off and stuck-on faults on PIPs, respectively; finally, columns four to six report the achieved fault coverage values for stuck-at, stuck-off and stuck-on faults, respectively.

It can be observed that the effectiveness of the proposed approach in achieving high fault coverage values scales well w.r.t. the size of the Area Under Test. Indeed, in all cases more than 98 % of faults are covered by the test bitstreams.

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

TABLE 2
Fault coverage achieved by OLT(RE)^2

Family	Device	#PW Sa	#PIP Soff	#PIP Son	SA	SOff	SOn
Virtex-4	FX12	38,784	444,476	427,792	38,784(100.00 %)	439,074 (98.78 %)	427.646 (99,97 %)
	FX100	111,179	1,317,736	1,270,955	111,179 (100.00 %)	1,305,778 (99.09 %)	1,248,119 (98.20 %)
Virtex-5	LX20T	79,425	941,323	900,205	79,225 (99.75 %)	929,720 (98.77 %)	887,975(98.64 %)
	LX330T	283,021	3,453,305	3,303,288	282,821 (99.93 %)	3,304,783 (95.70 %)	3,199,157 (96.85 %)
Virtex-6	CX130T	331,684	4,130,930	3,973,033	329,262 (99.27 %)	3,905,723 (94.55 %)	3,866,526 (97.32 %)
	LX760	937,398	11,728,174	11,253,950	932,617 (99.49 %)	10,819,240 (92.25 %)	10,648,488 (94.62 %)
Spartan-6	LX9	25,504	268,364	255,940	25,242 (98.97 %)	255,771 (95.30 %)	243,091 (94.98 %)
	LX150T	121,804	1,384,587	1,325,959	120,594 (99.01 %)	1,318,006 (95.19 %)	1,274,644 (96.13 %)

TABLE 3
Summary of the performance of OLT(RE)^2

Family	Device	# Testing circuits	Bitstream Size	Bitstream Size (Compr.)	Time
Spartan-6	LX9	4,179	98 MB	10 MB	3h:40m
	LX150T	16,328	914 MB	83 MB	40h:48m
Virtex-4	FX12	8,058	249 MB	31 MB	12h:52m
	FX100	38,245	1623 MB	149 MB	78h:25m
Virtex-5	LX20T	19,562	1217 MB	96 MB	47h:22m
	LX330T	34,081	1555 MB	150 MB	87h:46m
Virtex-6	CX130T	41,053	5418 MB	264 MB	95h:27m
	LX760	120,568	7364 MB	719 MB	382h:17m

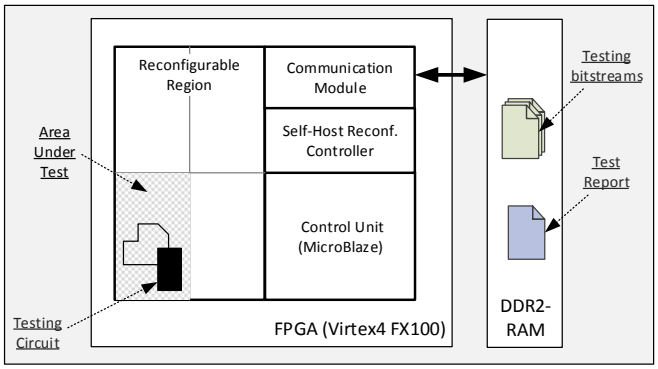

Fig. 13. The architecture of the considered reconfigurable module.

Finally, Table 5 reports the size of the Area Under Test in terms of number of clock regions (first column), number of test circuits generated to achieve the fault coverage values reported in Table 4 (second column) as well as the total size (in MB) of the test bitstream suite and the size of the test bitstream suite after bitstream compression (third and fourth columns, respectively) and, finally, the time required at design time to generate them.

Again, it can be observed that the proposed approach scales well both in terms of test bitstreams size and of test generation time w.r.t. the size of the Area Under Test. Given the results of the above presented analyses, we may conclude that OLT(RE)^2 can be effectively and efficiently used for a wide range of device families and that it does not suffer from the increase of the size of the area under test.

6.3 Runtime Performance Analysis

In order to demonstrate that the proposed testing approach can be applied in a real-world scenario on a complete reconfigurable system, we evaluated the run-time performance of the testing technique on the Dynamically Reconfigurable Processing Module (DRPM), a scalable FPGA-based prototyping environment for satellite payload-processing systems [1]. We were interested in both the time needed to transfer the test bitstream to the reconfigurable system and the time required to execute the entire test suite by partially reconfiguring the FPGA under test.

6.3.1 Experimental Platform

The DRPM combines a rad-hard System on Chip (the SpaceWire Remote Terminal Controller, based on a LEON2-FT), dynamically reconfigurable Xilinx FPGAs, and avionic interfaces, e.g., SpaceFibre and SpaceWire. In particular, the data processing modules host a Xilinx Virtex-4 FX100 FPGA (which provides dynamic partial reconfigurability). The overall architecture of the data processing module used in our experiment is shown in Figure 13. One of the key components is the self-hosting reconfiguration controller (SHRC) which interfaces the internal configuration access port (ICAP) and the FrameECC controller in the FPGA. Therefore, the SHRC is used to internally perform dynamic partial reconfiguration and to readback the configuration of the FPGA. A MicroBlaze CPU controls SHRC and initiates internal transfers from/to the DDR2 memory. The FPGA is partitioned in two main regions: a static region and a reconfigurable region. Within the static region, the main components are the Communication module, the SHRC and the MicroBlaze. The reconfigurable region is the target area of our test. Given the complexity and completeness of the DRPM, we believe that it represents the perfect run-time platform for our experiment.

The test is executed as follows: (i) the test bitstreams are sent from a host PC to the DDR2-RAM of the target systems; (ii) the MicroBlaze is triggered and executes partial dynamic reconfigurations in order to exhaustively place all the test circuits on the Area Under Test; (iii) the results of the execution of each test circuit are readback from the DRAM. Finally, when all the test circuits have been executed, a report is sent to the host PC.

6.3.2 Results Discussion

The execution of each test circuit on the DRPM took 30.281 ms, while the analysis of the results produced by each test took 1.373ms. Thus, the execution of the entire test set (38,245 test circuits) on the DRPM required about 20 minutes. We believe that this number is acceptable if we take into account that a reconfigurable system employed in a space mission does not execute frequent reconfigurations.

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

TABLE 4
Analysis of the scalability of OLT(RE)^2 on the Virtex-4 XC4VFX12 device (achieved fault coverage)

Area Under Test	#PW	#PIPs SOff	#PIPs SOn	SA	SOff	SOn
1 clock region	38,784	444,476	427,792	38,784 (100.00 %)	439,074 (98.78 %)	427,646 (99.97 %)
2 clock region	83,510	1,013,107	978,481	83,510 (100.00 %)	1,000,594 (98.76 %)	976,606 (99.81 %)
3 clock region	127,948	1,575,888	1,523,548	127,948 (100.00 %)	1,558,411 (98.89 %)	1,520,071 (99.77 %)
4 clock region	396,382	2,120,036	2,050,807	396,382 (100.00 %)	2,098,496 (98.98 %)	2,046,229 (99.78 %)

TABLE 5
Analysis of the scalability of OLT(RE)^2 on the Virtex-4 XC4VFX12 device (number and size of the test bitstreams and generation time)

RUT	#testCirc.	B. Size	B. Size (Compr.)	Time
1 clk reg.	8,058	249 MB	31 MB	12h:52m
2 clk reg.	19,014	999 MB	88 MB	34h:56m
3 clk reg.	30,552	1863 MB	148 MB	72h:07m
4 clk reg.	41,011	2912 MB	215 MB	102h:18m

Moreover, we estimated the time required to transfer the entire test suite from a hypothetical ground station to a hypothetical satellite hosting the DRPM: considering the bandwidth reported in [5] (306 Mbps) and considering the total size of the compressed suite reported in Table 3 (162 MB) we estimated a total transfer time of about 4 seconds, that, again is reasonable for a space mission.

7 Conclusions and Future Work

We have proposed OLT(RE)^2, a testing technique meant to be applied on-line and on-demand to detect permanent faults in reconfigurable systems. We believe that the proposed technique is particularly interesting for a twofold reason: on the one hand, it can help designers in making the use of state-of-the-art high-performance commercial-grade FPGAs viable for space applications; on the other hand, it could help in low-cost application scenarios where high-end radiation-hardened devices are not affordable. Experimental results have shown that the proposed approach may be applied to a large set of FPGA families and models, always allowing the great majority of faults both in physical wires and PIPs to be detected. Furthermore, OLT(RE)^2 has demonstrated to be highly scalable both in terms of execution time and of number and size of the test circuits when increasing the size of the Area Under Test. Finally, an experiment carried out on a test-bed reconfigurable system demonstrated that the time required to transfer and apply the test circuits makes the proposed approach viable for real-world space applications.

As future work we plan to introduce bitstream relocation and bitstream compression in the proposed flow to further decrease the size of the bitstream and thus the transfer time. Additionally, we aim at developing new test circuits with the goal of performing a more fine-grained fault diagnosis in case a fault is detected at run-time, thus making it possible to re-use partially faulty areas.

References

[1] L. Sterpone, M. Porrmann, and J. Hagemeyer, "A Novel Fault Tolerant and Runtime Reconfigurable Platform for Satellite Payload Processing," IEEE TRANSACTIONS ON COMPUTERS, vol. 62, no. 8, 2013.

[2] D. Sabena, L. Sterpone, M. Scholzel, T. Koal, H. Vierhaus, S. Wong, R. Glein, F. Rittner, C. Stender, M. Porrmann, and J. Hagemeyer, "Reconfigurable high performance architectures: How much are they ready for safety-critical applications?" in 19th IEEE European Test Symposium (ETS), May 2014, pp. 1–8.

[3] A. Paschalis, H. Michalik, N. Kranitis, C. Lopez-Ongil, and P. Reviriego Vasallo, "Dependable reconfigurable space systems: Challenges, new trends and case studies," in Proceedings of the IEEE 20th International On-Line Testing Symposium (IOLTS), July 2014, pp. 222–227.

[4] F. Bubenhagen, B. Fiethe, T. Lange, H. Michalik, and H. Michel, "Reconfigurable platforms for Data Processing on scientific space instruments," in Proceedings of the NASA/ESA Conference on Adaptive Hardware and Systems (AHS), June 2013, pp. 63–70.

[5] F. Rittner, R. Glein, T. Kolb, and B. Bernard, "Broadband FPGA payload processing in a harsh radiation environment," in Proceedings of the NASA/ESA Conference on Adaptive Hardware and Systems (AHS), July 2014, pp. 151–158.

[6] R. Ferguson and R. Tate, "Use of field programmable gate array technology in future space avionics," in Proceedings of the 24th Digital Avionics Systems Conference (DASC 2005), 2005.

[7] R. Baumann, "Radiation-induced Soft Errors in Advanced Semiconductor Technologies," IEEE Transactions on Device and Materials Reliability, vol. 5, no. 3, pp. 305 – 316, September 2005.

[8] J. Wang, "Radiation effects in FPGAs," in Proceedings of the 9th Workshop on Electronics for LHC Experiments, October 2003, pp. 34–43.

[9] P. Graham, M. Caffrey, J. Zimmerman, D. E. Johnson, P. Sundararajan, and C. Patterson, "Consequences and Categories of SRAM FPGA Configuration SEUs," in Proceedings of the 6th Military and Aerospace Applications of Programmable Logic Devices (MAPLD'03), September 2003.

[10] W. Huang, F. Meyer, N. Park, and F. Lombardi, "Testing Memory Modules in SRAM-based Configurable FPGAs," in Proceedings of the International Workshop on Memory Technology, Design and Testing, aug 1997, pp. 79 –86.

[11] M. Renovell, J. Portal, J. Figuras, and Y. Zorian, "Minimizing the Number of Test Configurations for Different FPGA Families," in Proceedings of the Eighth Asian Test Symposium (ATS '99), 1999, pp. 363 –368.

[12] J. Smith, T. Xia, and C. Stroud, "An Automated BIST Architecture for Testing and Diagnosing FPGA Interconnect Faults," Journal of Electronic Testing, vol. 22, pp. 239–253, 2006.

[13] M. Abramovici, C. Stroud, C. Hamilton, S. Wijesuriya, and V. Verma, "Using roving STARs for on-line testing and diagnosis of FPGAs in fault-tolerant applications," in Proceedings of the International Test Conference, 1999, pp. 973–982.

[14] M. Rozkovec, J. Jenicek, and O. Novak, "Application Dependent FPGA Testing Method," in Proceedings of the 13th Euromicro Conference on Digital System Design: Architectures, Methods and Tools (DSD10), sept. 2010, pp. 525 –530.

[15] M. Tahoori, "Application-Dependent Testing of FPGAs," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 14, no. 9, pp. 1024 –1033, 2006.

[16] C. Bernardeschi, L. Cassano, M. G. Cimino, and A. Domenici, "GABES: A genetic algorithm based environment for SEU testing in SRAM-FPGAs ," Journal of Systems Architecture, vol. 59, no. 10, Part D, pp. 1243 – 1254, 2013.

[17] L. Cassano, D. Cozzi, S. Korf, J. Hagemeyer, M. Porrmann, and L. Sterpone, "On-line testing of permanent radiation effects in reconfigurable systems," in Design, Automation Test in Europe Conference, March 2013, pp. 717–720.

[18] D. Sorrenti, D. Cozzi, S. Korf, L. Cassano, J. Hagemeyer, M. Porrmann, and C. Bernardeschi, "Exploiting dynamic partial reconfiguration for on-line on-demand testing of permanent faults in reconfigurable systems," in IEEE International Symposium on

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), Oct 2014, pp. 203–208.
[19] A. Cilardo, "New Techniques and Tools for Application-Dependent Testing of FPGA-Based Components," IEEE Transactions on Industrial Informatics, vol. 11, no. 1, pp. 94–103, Feb 2015.
[20] P. Roche, J.-L. Autran, G. Gasiot, and D. Munteanu, "Technology downscaling worsening radiation effects in bulk: SOI to the rescue," in IEEE International Electron Devices Meeting, Dec 2013, pp. 31.1.1–31.1.4.
[21] L. Bauer, C. Braun, M. Imhof, M. Kochte, E. Schneider, H. Zhang, J. Henkel, and H.-J. Wunderlich, "Test strategies for reliable runtime reconfigurable architectures," IEEE Transactions on Computers, vol. 62, no. 8, pp. 1494–1507, Aug 2013.
[22] C. Stroud, J. Nall, M. Lashinsky, and M. Abramovici, "BIST-based diagnosis of FPGA interconnect," in Proceedings of the International Test Conference, 2002, pp. 618–627.
[23] J. S and V. K. Agrawal, "Detection and Diagnosis of Faults in the Routing Resources of a SRAM based FPGAs," International Journal of Computer Applications, vol. 53, no. 13, pp. 18–22, September 2012, published by Foundation of Computer Science, New York, USA.
[24] X. Sun, P. Trouborst, J. Xu, and B. Chan, "Novel technique for built-in self-test of FPGA interconnects," in Proceedings of the IEEE International Test Conference. International Test Conference, 2000, pp. 795–795.
[25] J. Yao, B. Dixon, C. Stroud, and V. Nelson, "System-level Built-In Self-Test of global routing resources in Virtex-4 FPGAs," in Proceedings of the 41st Southeastern Symposium on System Theory, March 2009, pp. 29–32.
[26] M. Bellato, P. Bernardi, D. Bortolato, A. Candelori, M. Ceschia, A. Paccagnella, M. Rebaudengo, M. Reorda, M. Violante, and P. Zambolin, "Evaluating the effects of SEUs affecting the configuration memory of an SRAM-based FPGA," in Proceedings of the Design, Automation and Test in Europe Conference and Exhibition., vol. 1, Feb 2004, pp. 584–589 Vol.1.
[27] S. Korf, D. Cozzi, M. Koester, J. Hagemeyer, M. Porrmann, U. Ruckert, and M. Santambrogio, "Automatic HDL-Based Generation of Homogeneous Hard Macros for FPGAs," in Proceedings of th IEEE 19th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), May 2011, pp. 125–132.
[28] C. Beckhoff, D. Koch, and J. Torresen, "Portable module relocation and bitstream compression for Xilinx FPGAs," in 2014 24th International Conference on Field Programmable Logic and Applications (FPL), Sept 2014, pp. 1–8.
[29] T. Becker, W. Luk, and P. Cheung, "Enhancing Relocatability of Partial Bitstreams for Run-Time Reconfiguration," in Proceedings of the 15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, April 2007, pp. 35–44.
[30] H. Kalte, G. Lee, M. Porrmann, and U. Ruckert, "REPLICA: A Bitstream Manipulation Filter for Module Relocation in Partial Reconfigurable Systems," in Proceedings of the 19th IEEE International on Parallel and Distributed Processing Symposium, April 2005, pp. 151b–151b.

Sebastian Korf received the graduate degree as a Diplom-Ingenieur in electronical engineering combined with computer science from the University of Paderborn, Germany in 2011. He worked from 2012 to 2013 on self-optimizing systems in the group System and Circuit Technology at the Heinz Nixdorf Institute, University of Paderborn. Currently, he works on his PhD degree investigating the features of dynamic partial reconfiguration for the improvement of the reliability of SRAM-based FPGA systems in the research group Cognitronics and Sensor Systems, Center of Excellence Cognitive Interaction Technology, Bielefeld University.

Luca Cassano received the BS and MS and Ph.D. degrees in Computer Engineering from the University of Pisa, Italy. He is currently a Post-Doc researcher at the Politecnico di Milano, Italy. His research focuses on the use of formal methods and machine learning techniques for fault simulation, testing, untestability analysis, diagnosis and verification of digital circuits/systems. With his Ph.D. thesis, titled "Analysis and Test of the Effects of Single Event Upsets Affecting the Configuration Memory of SRAM-based FPGAs", he won the European semifinals of the 2014 TTTC's E. J. McCluskey Doctoral Thesis Award and he classified as runner-up at the world finals.

Jens Hagemeyer studied electrical engineering combined with computer science at the University of Paderborn, Germany. He received the diploma degree from the University of Paderborn, in 2006. After that he was with the Research Group System and Circuit Technology, University of Paderborn. In 2012, he joined the Cognitronics and Sensor Systems Group at Bielefeld University. He is working in the area of FPGA-centric system design, with a focus on large scale multi-FPGA systems. His current research interests include design of dynamically reconfigurable systems as well as novel, resource-efficient computer architectures.

Dario Cozzi received his Bachelor and Master degrees in Computer Engineering in 2007 and 2011, respectively, both from Politecnico di Milano, Italy. He is currently a Ph.D. student in the research group of Cognitronics and Sensor Systems, Center of Excellence Cognitive Interaction Technology, Bielefeld University, Germany. His research focuses on SRAM-based FPGAs used in space missions. In particular, he is working on partial dynamic reconfiguration and detection of permanent faults in SRAM-based FPGAs.

Andrea Domenici Andrea Domenici obtained his PhD in Information Engineering from the University of Pisa, Italy, in 1992 with a thesis on the implementation of the Gödel logic programming language on parallel machines. He is at the Department of Information Engineering of the University of Pisa, where he teaches Software Engineering and does research in the fields of dependable systems and application of formal methods in the development of safety- and mission-critical systems. He has also been active in Object-oriented design and Grid architectures.

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2586195, IEEE Transactions on Emerging Topics in Computing

Cinzia Bernardeschi received her Laurea degree and Ph.D. degree in computer science in 1987 and 1996 respectively, both from the University of Pisa. She is an associate professor with the Department of Information Engineering of the University of Pisa. Her research interests are in the area of software engineering, dependable systems and application of formal methods for specification and verification of safety-critical systems. Her most recent work is related to the application of theorem proving and model checking techniques for fault simulation and reliability analysis of electronic circuits and systems.

Luca Sterpone received the M.S. and Ph.D. degrees in computer engineering from Politecnico di Torino, Italy, in 2003 and 2007, respectively. He is currently an associate professor with the Department of Computer Engineering, Politecnico di Torino. His current research interests include design, validation, and test of safety-critical devices with particular emphasis on computer aided design tool for layout, synthesis, and place and route. He has been a Research Intern at Boeing Satellite Systems, El Segundo, CA, USA, and at EADS Innovation Works, Suresnes, France, during 2006 and 2007.

Mario Porrmann received the graduate degree as a Diplom-Ingenieur in electrical engineering from the University of Dortmund, Germany, in 1994, and the PhD degree in electrical engineering from the University of Paderborn, Germany, in 2001 for his work on performance evaluation of embedded neurocomputers. He is a senior lecturer (Akademischer Direktor) in the research group Cognitronics and Sensor Systems, Center of Excellence Cognitive Interaction Technology, Bielefeld University. From 2001 to 2009, he was Akademischer Oberrat and from 2010 to March 2012 acting professor of the research group System and Circuit Technology at the Heinz Nixdorf Institute, University of Paderborn. His scientific interests are in on-chip multiprocessor systems, dynamically reconfigurable hardware and resource-efficient computer architectures. He is member of the IEEE.

2168-6750 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.